

Proyecto Grupal 1

Diseño e Implementación de un ASIP para interpolación de imágenes

Fecha de asignación: 25 y 26 de sep de 2024
Grupos: 3 o 4 personas

Fecha de entrega: 21 de oct de 2024
Profesores: Jason Leitón Jiménez
Luis Barboza Artavia

Mediante el desarrollo de este proyecto, el estudiante aplicará los conceptos de arquitectura de computadores en el diseño e implementación en hardware de un *Application Specific Instruction Set Processor* (ASIP) para interpolar imágenes.

1. Descripción General

Para este proyecto se aplicarán los conceptos de arquitectura de computadoras para el diseño e implementación de un procesador vectorial. La arquitectura del set de instrucciones (ISA) será propuesto por cada grupo según las necesidades de la aplicación para implementar un algoritmo para interpolación de imágenes. El algoritmo que se implementará es interpolación bilineal, es recomendada para imágenes de escenarios con valores continuos.

La implementación de estos dos algoritmos se observan en la Figura 1.

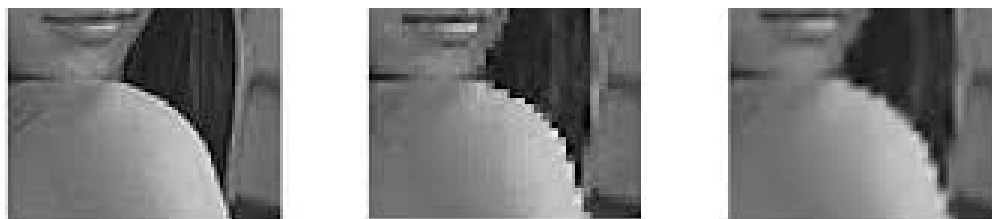


Figura 1: Implementación de algoritmos de interpolación (de izquierda a derecha): *groundtruth*, vecino más cercano y bilineal.

La implementación de este algoritmos se muestra a continuación para la imagen

$$I = \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix}, \quad (1)$$

donde cada valor de la matriz es un píxel.

1.1. Interpolación bilineal

El proceso de interpolación bilineal es un poco más complejo pues trata de rellenar el espacio con información continua, no replicando un mismo valor. Por ejemplo, el proceso de interpolación

bilineal para la imagen de la ecuación (1). Primero se comienza con los valores que ya se conocen.

1.1.1. Píxeles conocidos

Los píxeles que se usan de referencia son los marcados en **rojo**, los índices se muestran entre paréntesis en color **azul**

$$I' = \begin{bmatrix} 10(1) & a(2) & b(3) & 20(4) \\ c(5) & d(6) & e(7) & f(8) \\ g(9) & h(10) & i(11) & j(12) \\ 30(13) & k(14) & l(15) & 40(16) \end{bmatrix}. \quad (2)$$

El procedimiento para obtener los valores de la a a l . Se muestra a continuación.

1.1.2. Píxeles horizontales y verticales

Estos píxeles se calculan como si fuera una interpolación lineal. Se le da un mayor peso a los valores más cercanos al conocido. Los valores horizontales y verticales se muestran a continuación (son operaciones con enteros):

1. $a = \frac{4-2}{4-1} \times 10 + \frac{2-1}{4-1} \times 20 = \frac{2}{3} \times 10 + \frac{1}{3} \times 20 = 13$
2. $b = \frac{4-3}{4-1} \times 10 + \frac{3-1}{4-1} \times 20 = \frac{1}{3} \times 10 + \frac{2}{3} \times 20 = 17$
3. $c = \frac{13-5}{13-1} \times 10 + \frac{5-1}{13-1} \times 30 = \frac{2}{3} \times 10 + \frac{1}{3} \times 30 = 17$
4. $g = \frac{13-9}{13-1} \times 10 + \frac{9-1}{13-1} \times 30 = \frac{1}{3} \times 10 + \frac{2}{3} \times 30 = 23$
5. $k = \frac{16-14}{16-13} \times 30 + \frac{14-13}{16-13} \times 40 = \frac{2}{3} \times 30 + \frac{1}{3} \times 40 = 33$
6. $l = \frac{16-15}{16-13} \times 30 + \frac{15-13}{16-13} \times 40 = \frac{1}{3} \times 30 + \frac{2}{3} \times 40 = 37$
7. $f = \frac{16-8}{16-4} \times 30 + \frac{8-4}{16-4} \times 40 = \frac{2}{3} \times 20 + \frac{1}{3} \times 40 = 27$
8. $j = \frac{16-12}{16-4} \times 30 + \frac{12-4}{16-4} \times 40 = \frac{1}{3} \times 20 + \frac{2}{3} \times 40 = 33$

Para mayor guía se rellenan los valores faltantes en

$$I' = \begin{bmatrix} 10(1) & 13(2) & 17(3) & 20(4) \\ 17(5) & d(6) & e(7) & 27(8) \\ 23(9) & h(10) & i(11) & 33(12) \\ 30(13) & 33(14) & 37(15) & 40(16) \end{bmatrix}. \quad (3)$$

1.1.3. Píxeles intermedios

Estos píxeles se calculan usando los píxeles verticales y horizontales. Igual que el método anterior se le da mayor peso a los píxeles cercanos, en este caso los recién calculados en el paso anterior, como se observa en

$$I' = \begin{bmatrix} 10(1) & 13(2) & 17(3) & 20(4) \\ 17(5) & d(6) & e(7) & 27(8) \\ 23(9) & h(10) & i(11) & 33(12) \\ 30(13) & 33(14) & 37(15) & 40(16) \end{bmatrix}. \quad (4)$$

Para interpolar los píxeles d , e , h , i . Para estos valores se puede inferir ya sea horizontal o verticalmente, sin importar el eje se obtiene el mismo valor. En este caso se infiere horizontalmente:

1. $d = \frac{8-6}{8-5} \times 17 + \frac{6-5}{8-5} \times 27 = \frac{2}{3} \times 17 + \frac{1}{3} \times 27 = 20$
2. $e = \frac{8-7}{8-5} \times 17 + \frac{7-5}{8-5} \times 27 = \frac{1}{3} \times 17 + \frac{2}{3} \times 27 = 24$
3. $h = \frac{12-10}{12-9} \times 23 + \frac{10-9}{12-9} \times 33 = \frac{2}{3} \times 23 + \frac{1}{3} \times 33 = 26$
4. $i = \frac{12-11}{12-9} \times 23 + \frac{11-9}{12-9} \times 33 = \frac{1}{3} \times 23 + \frac{2}{3} \times 33 = 30$

La representación final de la matriz se observa en

$$I' = \begin{bmatrix} 10(1) & 13(2) & 17(3) & 20(4) \\ 17(5) & 20(6) & 24(7) & 27(8) \\ 23(9) & 26(10) & 30(11) & 33(12) \\ 30(13) & 33(14) & 37(15) & 40(16) \end{bmatrix} = \begin{bmatrix} 10 & 13 & 17 & 20 \\ 17 & 20 & 24 & 27 \\ 23 & 26 & 30 & 33 \\ 30 & 33 & 37 & 40 \end{bmatrix}. \quad (5)$$

Esta operación se realiza para cada cuadrante identificado.

1.2. Resultado de interpolación

Se observa que para la imagen

$$I = \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix}, \quad (6)$$

y para interpolación bilineal sería

$$I'' = \begin{bmatrix} 10 & 13 & 17 & 20 \\ 17 & 20 & 24 & 27 \\ 23 & 26 & 30 & 33 \\ 30 & 33 & 37 & 40 \end{bmatrix}. \quad (7)$$

2. Especificación

Se le solicita desarrollar una **arquitectura** y una **microarquitectura** que realice el proceso de ecualización de histogramas con una imagen libre con dimensión mínima de 390×390 .

En la salida se podrá seleccionar alguno de los siguientes cuadrantes mostrados en la Figura 2. Al cuadrante seleccionado se le aplicará la interpolación bilineal.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Figura 2: Cuadrícula de selección de la imagen de entrada

Se deben seguir los siguientes requisitos generales de funcionalidad:

1. El diseño completo debe poder ser sintetizable en una tarjeta de desarrollo Terasic DE1-SoC-M TL2 (debe caber todo ahí, inclusive la imagen de entrada y el producto de ecualización por histograma).
2. Las imágenes de entrada y salida deben ser almacenadas en memoria (se recomienda usar el bloque IP de la biblioteca de Quartus).
3. El sistema debe permitir la interacción con el usuario, para poder escoger el cuadrante mediante algún periférico (e.g., botones, switches).
4. La imagen resultante se mostrará por medio de VGA.
5. El ISA debe ser eficiente y congruente, con criterios de diseño definidos. Es importante hacer reuniones con el profesor para guía.
6. El formato de las imágenes será en escala de grises con píxeles con valores entre $[0, 255]$.
7. El sistema tiene que ser capaz de medir, almacenar en memoria y mostrar los ciclos por instrucción tomados para ejecución del código de asm. Esta lectura se realizará por medio de JTAG.

Requisitos de Arquitectura ISA:

1. Debe diseñar un conjunto de instrucciones y arquitectura que permita solucionar el problema planteado, considerando detalles como:
 - a) Modos de direccionamiento.
 - b) Tamaño y tipo de datos.
 - c) Tipo y sintaxis de las instrucciones.
 - d) Registros disponibles y sus nombres.
 - e) Codificación y descripción funcional de las instrucciones

Tome en cuenta que estos detalles deben ser justificados desde el punto de vista de diseño (complejidad, costo, área, recursos disponibles).
2. Las instrucciones a desarrollar son libres así como el tipo de datos. Aunque no hay un límite en cuanto la cantidad de instrucciones, es importante que provea al menos instrucciones para control de flujo, operaciones aritméticas-lógicas, acceso a memoria.
3. Los productos finales de esta etapa son el *instruction reference sheet* o *green sheet*.
4. El ISA debe ser personalizado y realizado por los estudiantes, **no se aceptarán ISAs ya diseñados** (e.g., ARM, x86, RISC-V, otros). Debe justificar cada característica del mismo.

Requisitos de Microarquitectura

1. La implementación diseñada debe ser correcta respecto a las reglas definidas por la arquitectura, esto quiere decir que el procesador debe ser capaz de ejecutar todas las instrucciones definidas y su especificación respecto a errores y excepciones.
2. El procesador diseñado debe emplear pipelining. Tenga en cuenta las implicaciones respecto a riesgos de dicha técnica, el uso de registros y unidades de ejecución. **No se revisará si no tiene pipeline.**
3. Debe ser implementado usando SystemVerilog.
4. **No se permite realizar módulos especializados de hardware.**
5. El procesador debe tener capacidad de segmentación de memoria en datos e instrucciones además debe ser capaz de acceder los dispositivos de entrada y salida del sistema (GPIO, volcado de memoria, switches, etc).

6. Cada unidad funcional del sistema debe ser debidamente probada en simulación, para verificar su funcionamiento correcto (unit tests). Además debe incluir pruebas de integración y sistema. Se le solicita un plan de pruebas donde especifique los objetivos y descripción de las pruebas junto con sus resultados.
7. Los resultados finales de esta etapa son:
 - a) El código fuente (SystemVerilog) y el bitstream para programar la tarjeta de desarrollo.
 - b) Un diagrama de bloques de la microarquitectura y descripción de las interacciones entre ellos.
 - c) Simulaciones de las pruebas unitarias y de integración.
8. Reporte de consumo de recursos del FPGA para el modelo.

Requisitos de Software:

1. Crear una aplicación (software) empleando la arquitectura diseñada, con el fin de implementar la interpolación de imágenes.
2. Debe realizar un programa ('compilador') que permita traducir las instrucciones del ISA a binario, con la finalidad de ejecutarlo en el procesador.

El proceso de diseño debe incluir propuestas y comparación de viabilidad de las mismas.

3. Evaluación y entregables

La defensa será el mismo día de la entrega y todos los archivos (incluyendo código fuente) serán entregados a las 11:59 pm ese mismo día. La evaluación del proyecto se da bajos los siguientes rubros contra rúbrica correspondiente:

- Presentación proyecto 100 % funcional (80 %) por medio de rúbrica.
- Artículo científico tipo *paper* (20 %): El paper a realizar deberá tener una extensión no mayor a 4 páginas completas (incluyendo bibliografías), deberá ser realizado con L^AT_EX, siguiendo un formato establecido (IEEE Transactions o ACM, por ejemplo). Se les provee un ejemplo de paper en el [enlace](#). En general el *paper* deberá contar con las siguientes secciones:

1. Abstract (en inglés): Un buen abstract tiene las siguientes características:

- a)* Un abstract permite a los lectores obtener la esencia o esencia de su artículo o artículo rápidamente, para decidir si leer el artículo completo.
 - b)* Un abstract prepara a los lectores para seguir la información detallada, los análisis y los argumentos en su artículo completo.
 - c)* Un abstract ayuda a los lectores a recordar puntos clave de su paper.
 - d)* Un abstract es de entre 150 y 250 palabras.
2. Palabras clave significativas (a lo sumo 6).
 3. Introducción: Una buena introducción muestra el contexto del problema o lo que se va a solucionar, introduce el tema al lector. Al final de la introducción se indica la organización del documento (primero se muestra el algoritmo, luego....).
 4. Sistema desarrollado.
 5. Resultados.
 6. Conclusiones escritas en prosa.
 7. Bibliografía, en formato IEEE y referenciadas en el texto (usar cite). Referencia bien para evitar problemas de plagio. Una documento no referenciado en el texto no existe.

Se seguirán los siguientes lineamientos:

1. Los documentos serán sometidos a control de plagios para eliminar cualquier intento de plagio con trabajos de semestres anteriores, actual o copias textuales, tendrán nota de cero los datos detectados. Se prohíbe el uso de referencias hacia sitios no confiables.
2. No coloque código fuente en los documentos, quita espacio y aporta poco. Mejor explique el código, páselo a pseudocódigo o use un diagrama.